[illegible]

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/607,167	06/27/2003	Hyong-Kyun Lee	P56833	5599

7590 10/05/2006

Robert E. Bushnell
Suite 300
1522 K Street, N.W.
Washington, DC 20005

EXAMINER	
ANYA, CHARLES E	
ART UNIT	**PAPER NUMBER**
2194	

DATE MAILED: 10/05/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)
	10/607,167	LEE, HYONG-KYUN
	Examiner	Art Unit
	Charles E. Anya	2194

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *15 October 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-22* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-22* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☒ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☒ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

[illegible]

WILLIAM THOMSON
SUPERVISORY PATENT EXAMINER

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *3/1/05; 3/15/06*.	6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-22 are pending in this application.

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

3. **Claim 3 is rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Pat. No. 6,223,134 B1 to Rust et al.**

4. As to claim 3, Rust teaches a method for commonly controlling device drivers, comprising the steps of: arranging a device independent access hierarchy between an application hierarchy and a device driver hierarchy (figure 5 (Class Drivers 304) Col. 13 Ln. 19 – 67); defining functions available in a corresponding device driver among functions of a function block in a function table (Col. 6 Ln. 34 – 51, IVI engine 306 Col. 16 Ln. 28 – 49, "...array of function names..." Col. 23 Ln. 29 – 42); when a device is initialized, allowing said device independent access hierarchy to generate a device handler identifier based on a standardized data format for said device and transmit the generated device handler identifier to the application hierarchy of a higher order ("...returns a handle..." Col. 20 Ln. 35 – 46, "...handle...returned..." Col. 23 Ln. 12 – 21,

"...returns a handle..." Col. 24 Ln. 22 – 29); and allowing the higher-order application hierarchy to call a predetermined device using the device handler identifier ("...this handle..." Col. 23 Ln. 12 – 28, "...use this handle..." Col. 24 Ln. 22 – 29), and allowing said device independent access hierarchy to identify a function of the corresponding device driver from the function table using the device handler identifier ("...pointer..." Col. 16 Ln. 28 – 49, "...this handle..." Col. 23 Ln. 12 – 28, Col. 24 Ln. 1 – 11) and call the function of the corresponding device driver ("...this handle..." Col. 23 Ln. 12 – 28, Col. 24 Ln. 22 – 29).

Claim Rejections - 35 USC § 103

5. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

6. Claims 1,2 and 12-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,223,134 B1 to Rust et al. in view of U.S. Pat. No. 6,993,772 B2 to Pike et al.

7. As to claim 1, Rust teaches a method for commonly controlling device drivers, comprising the steps of: arranging a device independent access hierarchy between an application hierarchy and a device driver hierarchy (figure 5 (Class Drivers 304) Col. 13

Ln. 19 – 67) and applying a standardized rule of said device independent access hierarchy to said application hierarchy and said device driver hierarchy (IVI Engine 306 Col. 16 Ln. 28 – 49); and allowing said application hierarchy to access the device driver hierarchy through the standardized rule of said device independent access hierarchy (Col. 6 Ln. 34 – 51, Class Drivers 304 Col. 16 Ln. 28 – 67).

Rust is silent with reference to allowing said device driver hierarchy to access said application hierarchy through the standardized rule of said device independent access hierarchy.

Pike teaches allowing said device driver hierarchy to access said application hierarchy through the standardized rule of said device independent access hierarchy (Instrument Engine 102 ("...sent and read to and from..." Col. 8 Ln. 3 – 14).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Pike and Rust because the teaching of Pike would improve the system of Rust by allowing communication between a user application and device driver such that a response could be returned to the user application.

8. As to claim 2, Pike teaches the method as set forth in claim 1, with said step of allowing said application hierarchy and said device driver hierarchy to access, comprising the steps of: allowing said application hierarchy to transmit control commands based on a standardized common format for a corresponding device driver to said device independent access hierarchy, and allowing said device independent

access hierarchy to convert the control commands into other control commands based on a local format and transmit the converted control commands to said device driver; and allowing said device driver to give a response to the converted control commands based on the local format to said device independent access hierarchy, and allowing the device independent access hierarchy to convert the response from said device driver into a response based on the standardized common format and transmit the response based on the standardized common format to said application hierarchy ("…translate(s)…" Col. 7 Ln. 25 – 55, Instrument Engine 102 ("…formats…sent and read to and from…" Col. 8 Ln. 3 – 14).

9. As to claim 12, Rust teaches a method, comprising: requesting loss of signal ("…check instrument status…") state information based on a standardized common format by an application to a device independent access hierarchy (figure 8A (Step 472) Col. 24 Ln. 32 – 37); the device independent access hierarchy allows the request from said application to provided in a first device local format ("…only required to have knowledge of the class driver 304…" Col. 15 Ln. 24 – 45) and requesting a first device driver to provide the loss of signal state information to said device independent access hierarchy (figure 8A (step 478) Col. 24 Ln. 43 – 45); responding to the request for loss of signal state information based on the first device local format (figure 8B (Step 487) Col. 23 – 27); responding to said application by said device independent access hierarchy for loss of signal state information based on the standardized common format (Check Status Callback Col. 36 Ln. 46 – 49, figure 25B (Step 928) Col. 40 Ln. 15 - 20).

Rust does not explicitly teach converting the request from said application into a first device local format.

Pike teaches converting the request from said application into a first device local format ("...translate(s)..." Col. 7 Ln. 25 – 55, Instrument Engine 102 ("...formats...sent and read to and from..." Col. 8 Ln. 3 – 14).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Pike and Rust because the teaching of Pike would improve the system of Rust by providing a powerful, concise mechanism that allows users to easily and quickly communicate details of different hardware interfaces having a distinct application program interface or driver modules (Pike Col. 3 Ln. 31 – 35).

10. As to claim 13, Rust teaches the method of claim 12, with said step of converting the request from said application further comprising of converting the request into a second device local format and requesting a second device driver to provide the loss of signal state information to said device independent access hierarchy based on the second device local format when a first device is converted to a second device and said first device driver is changed to said second device driver (Abstract ("...interchangeable instrument drivers), "...interchangeability..." Col. 6 Ln. 18 – 28, "...substitute other instruments..." Col. 7 Ln. 42 – 50, "...replace..." Col. 8 Ln. 22 – 28, Col. 15 Ln. 3 - 6).

11. As to claim 14, Rust teaches the method of claim 13, further comprising of converting control commands based on the standardized common format to control commands provided to the device drivers accommodating a change of said application to a second application without changing the control commands provided to the device drivers (Abstract "...interchangeable instrument drivers...").

12. As to claim 15, Rust teaches the method of claim 14, further comprised of providing a mutual interface between said application and said first and second device drivers by the device independent access hierarchy (Col. 13 Ln. 19 – 44); reading material from a device driver control block and accessing the first and second device drivers using predetermined functions (figure 5, Col. 16 Ln. 28 – 56, Col. 20 Ln. 1 – 46).

13. As to claim 16, Rust teaches the method of claim 15, further comprising of said device independent access hierarchy using device handler identifiers based on the standardized data format, said device handler identifiers corresponding to respective devices ("...handle..." Col. 20 Ln. 38 – 46, "...handle..." Col. 23 Ln. 12 – 21).

14. As to claim 17, Rust teaches the method of claim 16, further comprising: providing the device handler identifiers to said application from said device independent access hierarchy during an initialization of the corresponding device; and storing, by said application, the device handler identifiers and calling a corresponding device using

a corresponding device handler identifier ("...handle..." Col. 20 Ln. 38 – 46, "...handle..." Col. 23 Ln. 12 – 21).

15. As to claim 18, Rust teaches the method of claim 17, further comprising of said device independent access hierarchy determining according to said device handler identifier whether a certain device driver should be called and calling the certain device handler according to the determination ("...handle..." Col. 20 Ln. 38 – 46, "...handle..." Col. 23 Ln. 12 – 21).

16. As to claim 19, Rust the method of claim 18, with the device independent access hierarchy using certain pointers and function pointers in performing the standardized common format in the device independent access hierarchy (Col. 16 Ln. 22 – 49, Col. 20 Ln. 52 – 67, Col. 21 Ln. 17 – 30).

17. As to claim 20, Rust teaches the method of claim 19, further comprised of when said application is calling a function of a function block to be used, said device independent access hierarchy identifies the existence of a corresponding function from a function table and uses a device handler identifier to inform the initialization of the device driver accommodating said application to access a device driver using said device handler identifier ("...handle..." Col. 20 Ln. 38 – 46, "...handle..." Col. 23 Ln. 12 – 21).

18. As to claim 21, Rust teaches the method of claim 20, further comprised of not varying the device handler identifier value for the device when said first device driver is changed to said second device driver ("...handle..." Col. 20 Ln. 38 – 46, "...handle..." Col. 23 Ln. 12 – 21).

19. Claim 22 is rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,223,134 B1 to Rust et al. in view of U.S. Pat. No. 6,993,772 B2 to Pike et al. as applied to claim 21 above, and further in view of U.S. Pat. No. 5,802,365 to Kathail et al.

20. As to claim 22, Kathail teaches the method of claim 21, further comprising of varying the addresses of the pointers when said first device driver is changed to said second device driver (VerifyFragmentAsDriver Col. 31 Ln. 50 – 67).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Kathail, Pike and Rust because the teaching of Kathail would improve the system of Pike and Rust by providing a function for guaranteeing that there is a driver in a given fragment (Kathail Col. 31 Ln. 50 – 52).

21. Claims 4-11 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Pat. No. 6,223,134 B1 to Rust et al. in view of Applicant's Admitted Prior Art (hereinafter referred to as AAPA pages 11 and 17).

As to claim 4, AAPA teaches the method as set forth in claim 3, with said device handler identifier being represented as DCB handlerId[x1.x2.x3], where x1, x2 or x3 is an unsigned integer, x1 being a value of the level 1 meaning a device ID, x2 being a value of the level 2 meaning a logical or physical group number of a corresponding device, x3 being a value of a channel meaning a channel number of a corresponding device or group (page 17 lines 8 – 10).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of AAPA and Rust because the teaching of AAPA would improve the system of Rust by providing a data structure for storing device or peripheral information.

22. As to claim 5, AAPA teaches the method as set forth in claim 4, with values of x1, x2 and x3 being "0" corresponding to there being no corresponding level or channel and the value of x1 sequentially increasing from "1" when the device is initialized (page 17 lines 8 – 10).

23. As to claim 6, Rust teaches a method for commonly controlling device drivers, comprising the steps of: arranging a device independent access hierarchy between an application hierarchy and a device driver hierarchy (figure 5 (Class Drivers 304) Col. 13 Ln. 19 – 67); allowing said device independent access hierarchy to dynamically assign a device control block, containing elements for carrying out a standardized rule, corresponding to said device handler identifier; allowing said device independent

access hierarchy to provide said device handler identifier to said application hierarchy ("...returns handle..." Col. 20 Ln. 35 – 46, "...returns a handle..." Col. 24 Ln. 22 – 29); and allowing said application hierarchy to call a predetermined device through said device independent access hierarchy using said device handler identifier ("...this handle..." Col. 20 Ln. 35 – 46, "...use this handle..." Col. 24 Ln. 22 – 29).

Rust is silent with reference to when a device initialization is controlled by said application hierarchy, allowing said device independent access hierarchy to carry out level 1 initialization, level 2 initialization and channel initialization and generate a device handler identifier based on a standardized data format for a device;

AAPA teaches when a device initialization is controlled by said application hierarchy, allowing said device independent access hierarchy to carry out level 1 initialization, level 2 initialization and channel initialization and generate a device handler identifier based on a standardized data format for a device (page 17 lines 8 – 10).

As to claim 7, AAPA teaches the method as set forth in claim 6, with the elements of said device control block comprising a pointer of "*pControlTable" for pointing a position of a command control table, the command control table containing a command identifier having a standardized unique value and a command function pointer mapped to the command identifier, a pointer of "*pDDCB" for pointing a position of a device driver control table through which the existence and position of a corresponding function is identified, and a pointer "*pAnchor" for pointing a next level (page 17 lines 8 – 10: NOTE: when level 1 is initialized, the DCB including "*pControlTable","*pDDCB" and "*pAnchor" are provided).

24. As to claim 8, AAPA teaches the method as set forth in claim 6, with the elements of said device control block comprising a pointer of "*pHandler" for pointing a position of a given initialization profile when a device is initialized, a function pointer of "*fpInitDevice" being used when a device is initialized, a function pointer of "*fpOpenChannel" being used when a channel is open, a function pointer of "*fpCloseChannel" being used when a channel is closed, a function pointer of "*fpRead" being used when data of an open channel is read, a function pointer of "*fpWrite" being used when data of the open channel is written, a function pointer of "*fpReset" being used when a device is reset, a pointer of "*pControlTable" for pointing a position of a command control table containing a command identifier having a standardized unique value and a command function pointer mapped to the command identifier, a pointer of "*pDDCB" for pointing a position of a device driver control table through which the existence and position of a corresponding function is identified, a pointer of "*pEventTable" for pointing a position of an event table, and a pointer "*pAnchor" for pointing a next level (page 17 lines 8 – 10: NOTE: when level 1 is initialized, the DCB including "*pHandler", "*fpInitDevice", "*fpOpenChannel", "*fpCloseChannel", "*fpRead", "*fpWrite", "*fpReset", "*pControlTable","*pDDCB" and "*pAnchor" are provided)..

25. As to claim 9, AAPA teaches the method as set forth in claim 6, with the level 1 initialization of said device being made by giving a device identifier value of x1 as a unique value for each device based on a sequence of the level 1 initialization in the device handler identifier represented as DCB handlerId[x1.x2.x3] where x1, x2 or x3 is an unsigned integer (page 17 lines 8 – 10).

26. As to claim 10, AAPA teaches the method as set forth in claim 9, with the level 2 initialization of the device being made by referring to the number of logical or physical groups, assigning anchors, and giving a group value of x2 as a unique value for each anchor in the device handler identifier represented as DCB handlerId[x1.x2.x3] where x1, x2 or x3 is an unsigned integer (page 17 lines 8 – 10).

27. As to claim 11, AAPA teaches the method as set forth in claim 10, with the level 3 initialization of the device being made by giving a channel value of x3 for each of channels belonging to the device and groups within the device on the basis of an open channel sequence in the device handler identifier represented as DCB handlerId[x1.x2.x3] where x1, x2 or x3 is an unsigned integer (page 17 lines 8 – 10).

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Charles E. Anya whose telephone number is (571) 272-3757. The examiner can normally be reached on M-F (8:30-5:00).

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Thomson can be reached on (571) 272-3718. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Charles E Anya
Examiner
Art Unit 2194

cea.

[illegible]

WILLIAM THOMSON
SUPERVISORY PATENT EXAMINER